# Foundations for Survivable Systems Engineering

Dr. Robert Ellison, Richard Linger, Dr. Howard Lipson, Dr. Nancy Mead, and Andrew Moore
*Software Engineering Institute*

*The complexity of today's large-scale networked systems increases their vulnerability to intrusion, compromise, and failure. We are addressing the survivability of these systems by establishing new methods for risk assessment and by developing engineering technologies for analysis and design of survivable systems.*

Survivability of critical infrastructure systems has become an urgent priority. These large-scale networked systems improve the efficiency of organizations through new levels of integration and communication. However, increased integration is accompanied by increased risks of intrusion, compromise, and cascade failure effects. Incorporating survivability into these systems can mitigate these risks.

Survivability focuses on preserving essential services, even when systems are penetrated and compromised [1]. As an emerging discipline, survivability builds on related fields of study (e.g., security, fault tolerance, safety, reliability, reuse, verification, and testing) and introduces new concepts and principles.

Survivability is defined as "the capability of a system to fulfill its mission, in a timely manner, in the presence of attacks, failures, or accidents" [2]. The term *system* is used in the broadest possible sense to include networks and large-scale systems. A key observation in survivability engineering is that no amount of security can guarantee that systems will not be penetrated and compromised. The complexities of Web-based services, issues of function and quality in commercial off-the-shelf (COTS) usage, and the proliferation of end-user devices and channels, combined with the growing sophistication of attacks and intrusions, present formidable engineering challenges in survivable system analysis and development.

Attacks exploit not only specific system vulnerabilities but also trust relations between systems. Attacks can target networks, devices, and user task flows. Sophisticated intruders include cyber-terrorists, non-state activists, and state-sponsored adversaries (foreign intelligence services and militaries), as well as insiders. Sophisticated intrusions are becoming more likely and more difficult to counter.

Many attacks target vulnerabilities in system components such as domain name servers or Web servers. Boundary control mechanisms, such as firewalls and demilitarized zones, provide some defense against these attacks. But it is often the case that security is addressed too late in the development cycle, with boundary controllers used for after-the-fact remediation when systems are deployed. Moreover, the adequacy of boundary controllers decreases as user task flows traverse multiple system boundaries and security administration domains. Sophisticated intruders can attack a broad range of targets across domains. Resistance and response to such attacks are often the responsibility of multiple enterprises and their system and application architectures.

> *"A key observation in survivability engineering is that no amount of security can guarantee that systems will not be penetrated and compromised."*

Security typically focuses on what is regarded as well-defined boundaries and control of internal components and systems within those perimeters. The reality of today's large-scale network systems is quite different. User task flows, system boundaries, and user communities are dynamic and difficult to analyze. The question of where or how to define the system boundary becomes highly important when considering survivability. The old notions of system boundaries may not fit the current environment.

Web services, although seemingly innocuous, may provide an opportunity for an attack. Remote access to systems such as that afforded by cable modem connections may also enable attacks. Any facility that provides an opportunity for attack on your system should be considered when performing a survivability analysis. Task flows cross multiple system and organizational boundaries and exhibit dependencies on external systems and on COTS components. New Web service and network communication infrastructures support such flows. And open-distributed architectures present whole new categories of vulnerabilities. These system realities drive two key problems in survivability design and development:

- How to design survivability into highly distributed systems despite limited central administration, poor visibility of end-to-end task flows and system dependencies, and dynamic functionality and usage.
- How to manage survivable system evolution in terms of changes in functional requirements, threats, and operating environments.

## The CERT Survivable Systems Research Agenda

We believe that new engineering methods are required to deal with these problems within the realities of today's dynamic, network-centric systems. Our research is aimed at theoretical foundations, language representations, and rigorous yet practical unified engineering methods to represent and reason about systems, their (often COTS) components, and their threat environments. Much of our work is documented in publications that can be found and downloaded from the CERT Web site at <www.cert.org>, particularly in the pages on "Survivable Systems Engineering" at <www.cert.org/sna>. Our overall objective is to improve system engineering practices for survivability. Such practices require solid engineering foundations.

For each life cycle activity, survivability goals should be addressed and methods to ensure survivability incorporated. If addressed at all, survivability issues are often relegated to a separate thread of project activity, with the result that survivability is treated as an add-on property. This isolation of survivability considerations from primary system development tasks results in an unfortunate separation of concerns. Survivability should be integrated and treated on a par with other system properties to develop systems with required functionality and performance that can also withstand failures and compromises. A survivability baseline needs to be established fairly early on, for example, during the development of concept of operations, and revisited at major development milestones such as requirements baseline, architecture baseline, etc. This sounds as if it suggests a waterfall-type life cycle, but in fact works nicely with more modern life cycle models such as the spiral model.

In some cases, existing development methods can enhance survivability. Current research is creating new methods that can be applied; however, more research and experimentation is required before the goal of survivability can become a reality. Our research agenda has its roots in the CERT (formerly known as Computer Emergency Response Team) Survivable Systems Analysis (SSA) method (formerly called Survivable Network Analysis) that we have been applying with clients for several years. Although we do not have documented cost/benefit data, in most cases it is clear to the clients that our recommendations will improve the survivability of their systems, thus the implementation decision is relatively easy to make.

SSA is a structured engineering process aimed at improving survivability characteristics of new or existing systems. A small team of survivability experts working with a client team of subject matter experts conducts it. SSA is carried out in a series of joint working sessions, and the findings are summarized in a report for management action [3]. The SSA process begins with briefings from system users, stakeholders, and developers typically focused at the architecture level. The discovery process continues with developer, user, and stakeholder views of essential services and assets of the system, that is, the services and assets that must be available no matter what the threat environment and state of compromise. These services are formulated as stepwise usage scenarios and traced through the architecture to reveal corresponding essential components.

Next, representative intrusions are identified based on analysis of the threat environment and, likewise, expressed as usage scenarios for tracing through the architecture to reveal components that can be compromised. With this information, it is possible to identify soft spot components that are both essential and able to be compromised, followed by survivability analysis for improvements to resistance, recognition, and recovery strategies within the system architecture. It is often the case that recommendations propagate to areas such as requirements, policy, and operations. Our application of SSA with clients has resulted in three key observations that drive the research agenda:

- Systematic evaluation methods are required for assessing COTS component survivability. Many organizations are developing mission-critical systems using COTS components. COTS can offer lower, up-front costs than custom-built solutions, but acquiring organizations lack access to the artifacts of the software engineering process used to create the components. Analysis of engineering artifacts is the traditional means for verifying the survivability of custom-built systems. One way to partially compensate for this lack of access is to use a vendor-risk assessment as a tool in building, maintaining, and evolving survivable systems. We are developing a risk-management approach called Vendor Risk Assessment and Threat Evaluation (V-RATE) [4] for assessing the survivability of COTS-based systems. V-RATE assessment helps acquiring organizations to understand the trade-offs associated with using COTS products, and to achieve the required assurance levels through evaluation and interaction with COTS vendors. It also supports comparison of different system designs based on alternative COTS products.
- Large-scale network system complexities can be reduced and managed by a unified engineering discipline for analysis and design that includes survivability in a comprehensive framework. Complexities of large-scale network system analysis and design often exceed engineering capabilities for intellectual control. We are defining engineering foundations for Flow-Service-Quality (FSQ) technology [5] based on user task flow structures and their architecture traces, a computational approach to quality attributes (including survivability), and an architecture framework for dynamic management of flows and their quality attributes. This process can be applied to specification, design, and operation of new systems, as well as to analysis of existing systems for survivability dependencies and risks that can impact mission performance. It also assists in integrating stovepipe systems to support new mission objectives.
- Structured documentation and systematic use of attack patterns and survivability strategies can help design and analyze intrusion-resistant architectures. Major investment in information security technology by a business or military enterprise often translates into little, or questionable, value to the operational mission. A primary reason is that many design and analysis efforts focus on deciding which popular security technologies to integrate, rather than on a rational assessment of how to address attacks that are likely to compromise the mission. Our work involves incorporating intrusion and risk-analysis techniques into existing development practices. This work requires consideration of the larger operational context in which system technology resides, which we call the *enterprise*. Enterprise architectures need to be developed and analyzed just like the systems on which they are based. These research projects are discussed in detail below.

> ***"A survivability baseline needs to be established fairly early on ... during the development of concept of operations and revisited at major development milestones."***

## Vendor Risk Assessment and Threat Evaluation Project

Building survivable systems using COTS components is a daunting task because the developer has little or no access to the artifacts of the software engineering process used to create the components. These artifacts are the primary sources from which assurance evidence for a composite system is derived. One way to partially compensate is to use vendor risk assessments as a tool to help build, maintain, and evolve survivable systems. Such an assessment can be used as a new source of assurance evidence of a system's survivability.

Our vendor risk assessment approach, V-RATE, is based on the taxonomy described in Table 1. Two broad categories are at the highest level of the taxonomy: 1) vendor-inherent risk elements, and 2) vendor-risk elements that are associated with your own risk management skills. The output of an assessment based on the V-RATE taxonomy is a vendor-risk profile for the system being evaluated. We envision a large and growing collection of ven-

| Vendor's Inherent Risk Elements | |
|---|---|
| Visibility of Product Attributes | Openness - degree of visibility into design and engineering processes.<br>Independent testing organizations. |
| Technical Competence | Survivability capability maturity.<br>Existence of vendor ratings/certifications.<br>Evidence of adherence to applicable industry standards and government regulations.<br>Demonstrated diversity and redundancy in a vendor's products and services.<br>Existence of a vendor team that deals effectively with security/survivability issues. |
| Performance History | Evidence that demonstrates a track record of dealing successfully or unsuccessfully with survivability issues and events. |
| Compliance | Responsiveness to security/survivability issues (which can include related quality issues such as reliability, performance, safety, and usability).<br>Responsiveness to requests for new features and improvements.<br>Willingness to cooperate with third-party testers and certifiers. |
| Trustworthiness | Track record/word-of-mouth.<br>Evidence of skill at evaluating trustworthiness of personnel, e.g., the vendor consistently checks the character references of new hires and periodically re-checks all personnel. |
| Business Management Competence | Economic viability.<br>Vendor's risk management skills in dealing with subcontractors. |
| Controlled Evolution | Clearly specified (or discernible) evolutionary path.<br>Product integration stability.<br>Product evolution supports continual survivability improvement. |
| **Vendor Risk Elements Associated With Your Risk Management Skills in Dealing With Vendors** | |
| Technical Risk-Mitigating Factors | Your skill at evaluating a product's quality attributes (in particular, those quality attributes that can contribute to system survivability such as security, reliability, performance, safety, and usability).<br>Your skill at evaluating vendor technical competence.<br>Your awareness of existing vendor ratings and certifications.<br>Demonstrated diversity and redundancy in the integration of vendor products and services.<br>Use of architectural tools and techniques (e.g., wrappers) to limit risks associated with a vendor product.<br>Your association with expert security/survivability organizations and the existence of a dedicated security/survivability group within your own organization. |
| Nontechnical Mitigation of Risk | Legal (e.g. license agreements).<br>Economic (e.g. insurance).<br>Political and social (e.g. regulatory protection). |
| Independence/Interdependence | You examine the vendor products and services associated with your system and look for interdependencies that could threaten survivability. |
| Your Exposure | You determine what elements of your system are dependent upon the competence, trustworthiness, and thoroughness of the vendor. |
| Mission Alignment/ Vendor Compatibility | You evaluate the alignment of your mission and the required software quality attributes (SQAs) with the vendor's mission and SQAs. |
| Your Negotiating Skill/ Bargaining Power | Use of economic or other leverage to obtain vendor concessions that enhance survivability such as early notification of security vulnerabilities. |

Table 1: *The V-RATE Taxonomy*

dor-risk profiles tied to real-world performance histories, providing empirical data against which a newly generated risk profile can be compared. A vendor-risk profile can be used to assess the risk associated with the use of a product in a particular threat environment and to identify areas for additional risk-mitigation activities. Because a single numerical rating would not provide sufficient guidance for these risk-mitigation activities, the vendor-risk profile helps identify your risks in each of the V-RATE taxonomy areas and allows you to consider your risk tolerance with respect to each element of the taxonomy.

We need to apply the V-RATE method to real-world, mission-critical systems. Such case studies will help us fine tune and validate the method and demonstrate its use within a realistic life cycle process. These studies will also help us to understand the risks associated with using COTS components for specific system missions. Details of the application of V-RATE (such as the specific evidence that needs to be gathered) may differ for different domains (e.g., military mission-critical, e-commerce, and financial systems). Since survivability is heavily dependent upon the context of the mission, understanding these differences is critical to V-RATE's successful application. We have an immediate plan for conducting a case study of the V-RATE method with a Carnegie Mellon University project in the coming months.

## Flow-Service-Quality Engineering Project

Imagine the flow of communications and operations among networked systems that support the simple task of purchasing gasoline with a credit card. The purchaser must enter input data. Communications must be established with the credit card organization, perhaps through a combination of land lines and satellite links. Credit databases and business rule services must be accessed, perhaps on multiple platforms, and results must be transmitted back to the pump, all in a few seconds. Of course, other customers are likely invoking the same flow from pumps across the country at the same time.

This flow of operations crosses multiple system boundaries and combines user inputs and the results of many system service uses along the way, all to satisfy the mission objective of purchasing gasoline. In more general terms, a flow begins with a mission objective (purchase gasoline) and elaborates into a sequence of user tasks (enter data, select the product, etc.). This turns into a traversal of a complex network to locate and execute the system services (databases, business rules, etc.) required to satisfy the mission.

From an engineering viewpoint, it is easy to see that such a flow represents a specification that a system design must satisfy, and that the design must accommodate the different types and volumes of flows that its many users require. In operation, such a system must typically satisfy hundreds or thousands of such flows simultaneously. Flows must also satisfy required quality attributes such as reliability, security, and survivability. Because flows cross many security domains in multiple systems, there are many opportunities for intrusion and compromise that can impact security and survivability. If a gasoline purchase flow is compromised, it is an inconvenience. But if a flow linking sensors and weapons in a complex battle management system is compromised, it is an entirely different matter. So it is worth investigating flows and their properties to better understand security and survivability issues in complex networked systems.

Modern enterprises are irreversibly dependent on large-scale networked systems. Unfortunately, the complexity of these systems frequently exceeds current engineering capabilities for intellectual control, resulting in persistent difficulties in acquisition, development, management, and evolution. These systems exhibit indeterminate boundaries, ever-changing linkages to other (often stovepipe) systems, COTS capability and quality uncertainties, dynamic function and usage, and continual requirements for evolution. Complexity is compounded by extensive asynchronous behavior, that is, simultaneous shared use of system services by multiple users that results in a virtually unknowable interleaving of operations and communications among system components.

A central issue in modern system development is how to maintain intellectual control over such complex structures and the asynchronous behaviors they produce. In short, what are the stable and dependable anchors for specification and design that can provide a unified engineering discipline for large-scale network system acquisition and development? We believe that FSQ engineering can provide that discipline [5].

In complex network systems with constantly varying function and usage, flows and their corresponding architecture traversals of system services can serve as the sought-after stable foundations for functional and nonfunctional (quality attribute) specification and intellectual control. The objective of our FSQ research is to provide engineering methods to represent and reason about system flows as essential artifacts of complex system analysis and development. System flows are composed of system services and must satisfy quality attributes such as reliability, performance, and survivability. Therefore, it is these three first-class concepts, *flow*, *service*, and *quality* that form the basis of the FSQ framework for engineering large-scale network systems.

Flows can be expressed in virtually any language using flow structure templates that permit precise specification of mission objectives, corresponding user tasks, and refinements into traversals of system services. In execution, services invoked by flows typically experience a blizzard of asynchronous usage interleavings that defy human understanding. A key result of our research is an approach to flow definition that guarantees it can be expressed in simple procedural structures for straightforward human understanding and analysis, despite the underlying asynchronous behavior of its service uses.

These procedural structures embody nested and sequenced service invocations expressed in terms of ordinary sequence, alternation, iteration, and concurrent structures. Such structures enable precise refinement, abstraction, and verification of flows for human understanding. In addition, flows can be organized into related flowsets associated with particular missions and

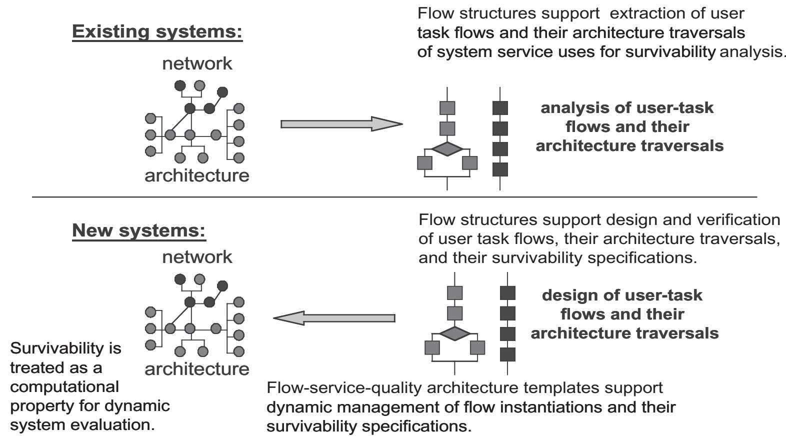


Figure 1: *Flow-Service-Quality Engineering Operations*

network components, and a rich set of operations can be applied to flow correlation and dependency analysis and simulation, of particular value for integrating existing stovepipe systems. Flows also define required levels of quality attributes for themselves, as well as for execution of the services they reference. FSQ engineering operations for existing and new systems are depicted in Figure 1.

In FSQ engineering, quality attributes such as security, survivability, reliability, and availability are defined as computational functions and are associated with both flows and services. Substantial effort has been devoted in the past to development of descriptive and often subjective a priori characterizations of the quality attributes of systems. Rather than focusing on descriptive predictions of limited value for dynamic networks, we adopt an alternate approach and ask how such attributes can be defined, computed, and acted upon as dynamic characteristics of system operation. That is, we wish to define quality attributes as functions to be computed, rather than as static estimates of capabilities.

While such functions rely on what can be computed and may differ thereby from traditional views of quality attributes, they can permit new approaches to attribute analysis, design, and operational evaluation. A key aspect of the computational approach is the ability to associate quality attributes with specific flows rather than with entire systems, thereby permitting differentiation among attribute capabilities based on mission criticality in survivability engineering.

In a world of flow-centric engineering and computational quality attributes, it is natural to consider system architecture templates based on dynamic flow and quality attribute management. We are investigating such FSQ architectures as straightforward implementations of flow-based systems.

Flow-structure engineering can reduce complexity and add clarity to the development of key system artifacts. First, flow specifications of enterprise tasks can be designed and verified with full human understanding (at various levels of abstraction in a rigorous and seamless process) from mission requirements down to architectural components. Second, a specification of network system behavior is defined as the set of flows of its service uses. And third, the specification of each service in a network system incorporates all its uses in all the flows wherein it appears.

Flow structures prescribe dynamic network linkages and operations, define composition requirements among nodes and services, and support both centralized and distributed control. Flow structures have the potential to reduce complexity and improve manageability in network system acquisition, development, management, and operation and can contribute to integration of diverse stovepipe systems to meet new mission requirements. In addition, flow structures can be used to extract and document mission-critical operations in existing systems to better understand component dependencies for survivability analysis.

## Intrusion-Aware Design Project

Developers in many engineering disciplines rely on engineering failure data to improve their designs and methods. Imagine the result if bridge builders had ignored the lessons learned from the torsional oscillations that caused the Tacoma Narrows Bridge to collapse. Or, if ship builders had ignored the lessons learned about inadequate lifeboat space and manning that allowed the great loss of life when the Titanic sank. Engineering success requires that we also learn from the less famous disasters. The aerospace community, for example, has institutionalized a means for learning from air traffic accidents that has resulted in a very low risk of death during

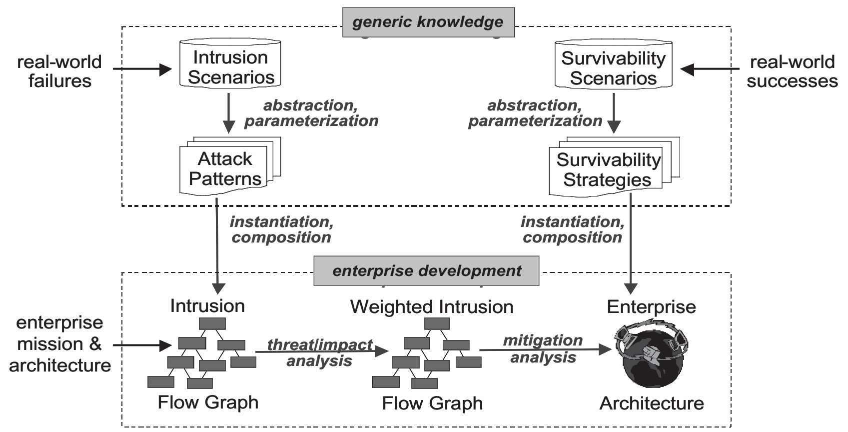


Figure 2: *Intrusion-Aware Design Refinement*

air travel, despite its inherent hazards. Successful architects design structures to survive known faults in building materials, construction methods, and the environment.

Unfortunately, information system developers generally do not use security failure or attack data to improve the security and survivability of systems that they develop. Information systems being built and managed today are prone to the same or similar vulnerabilities that have plagued them for years. In addition, increasingly sophisticated attacks exploit these vulnerabilities at an alarming rate. As seen by recent Internet worms and viruses released (e.g., Melissa, Love Letter, Code Red, Nimda), attackers share tools and knowledge to amplify their capability. The increasingly sophisticated tools currently available permit relatively inexperienced individuals to execute very sophisticated attacks. We have seen such attacks escalate with the intensity of political conflicts such as the war in Kosovo, the tensions between the United States and China, and the conflict between India and Pakistan [6]. While such attacks are often in the form of embarrassing Web site defacements, attackers are starting to target the perceptions of users, such as attempts to modify the content of major news publications or company press releases. In general, attacks can target a system's internal users and (COTS) components as well as external trusted systems and user communities.

Businesses and governments have historically been reluctant to disclose information about attacks on their systems for fear of losing public confidence or fear that other attackers would exploit the same or similar vulnerabilities. However, increased public interest and media coverage of the Internet's security problems has resulted in increased publication of attack data in books, Internet newsgroups, and CERT security advisories, for example. Much of the available attack information is very detailed in terms of software versions, enterprise-specific configurations, and attacker-specific scripts. Such details have a relatively short life as the attackers create and revise their tools and methods. However, the general patterns of attack are much less variable over time. Attack patterns describe general attack strategies, such as the various forms of denial-of-service attacks, and can be structured so they can be applied in a variety of contexts.

Intrusion-aware design methods enable information system engineers to use attack patterns in a structured way to improve information system security and survivability. Our approach is to collect as much knowledge about attack patterns and survivability strategies as possible to support the development and analysis of specific enterprises. Such a knowledge base can assist in identifying the general system risks and the most appropriate mitigation techniques. For example, network-based denial-of-service attacks suggest the need to distribute and diversify critical services, provide spare capacity, and/or attempt intruder trace-back, filtering, and possible apprehension.

Attack and survivability information needs to be structured and reusable so they can be applied in the iterative refinement of survivability architectures. By building the knowledge base so that it is independent of the enterprise, we provide a means for building enterprise-specific intrusion flow graphs in an affordable way, thus making the iterative refinement and analysis of the enterprise architecture cost-effective.

As shown in Figure 2, we build intrusion scenarios from real-world failures documented in, for example, incident and vulnerability databases. This effort requires fusing sometimes low-level incident data together to understand and describe larger-scale intrusions. We interpret intrusions broadly to include attacks that target people and task flows as well as those that target technology. We develop a means to derive commonly recurring attack patterns from intrusion scenarios. These attack patterns are parameterized so that they can be instantiated for varying enterprise environments. Enterprise-specific intrusion flow graphs are generated from these attack patterns through an instantiation and composition process [7].

Risk analysis techniques are used to prioritize the intrusions through threat and impact analyses. Mitigation analysis of these intrusions helps identify relevant survivability strategies that are used to refine the enterprise architecture in the most beneficial directions. The survivability strategies are derived from real-world survivability scenarios documented through years of practical experience in the area.

Intrusion-aware design does not reinvent risk analysis, but uses and augments risk analysis and management techniques where helpful. Our near-term focus is to explore the viability of this approach through its application to improve security and survivability of a particular enterprise architecture for a particular class of attacks. With evidence of the method's efficacy, our efforts will shift to developing and structuring the generic knowledge for intrusion and survivability scenario analysis (top part of Figure 2). Showing how to use this generic knowledge with existing risk management techniques for intrusion analysis and architecture improvement (bottom part Figure 2) is also a focus and key to the success of the approach. The key benefits of the approach are as follows:

- More structured/systematic means to document enterprise threats.
- Better understanding of enterprise mission vulnerability to sophisticated, multi-stage attacks.
- Improved accuracy and speed of risk analysis and management activities.
- Improved ability to identify architectural strategies to counter likely, high-consequence attacks.
- Faster, iterated improvement to enterprise architecture and overall survivability.

Successful application of intrusion-aware design methods should lead to enterprise architectures that demonstrably tolerate sophisticated attacks, providing higher confidence that the enterprise successfully carries out its mission.◆

## Acknowledgements

It is a pleasure to acknowledge contributions to this work by Dr. Alan Hevner, Dr. Mark Pleszkoch, Dr. Carol Sledge, Dr. Ann Sobel, and Dr. Gwen Walton.

## References

1. Anderson, R. H., A. C. Hearn, and R. O. Hundley. "RAND Studies of Cyberspace Security Issues and the Concept of a U.S. Minimum Essential Information Infrastructure." Proceedings of the 1997 Information Survivability Workshop. IEEE Computer Society, San Diego, Calif., 12-13 Feb. 1997. Available at <www.cert.org/research/isw/isw97/front_page.html>.
2. Ellison, R., D. Fisher, R. C. Linger, H. F. Lipson, T. Longstaff, and N. R. Mead. Survivable Network Systems: An Emerging Discipline (CMU/SEI-97-TR-013, ADA341963). Pittsburgh, Penn.: Software Engineering Institute, Carnegie Mellon University, Nov. 1997, revised May 1999. Available at <http://www.cert.org/research>.
3. Mead, N. R., R. J. Ellison, R. C. Linger, T. A. Longstaff, and J. McHugh. Survivable Network Analysis Method (CMU/SEI-2000-TR-013). Pittsburgh, Penn.: Software Engineering Institute, Carnegie Mellon University, 2000. Available at <www.sei.cmu.edu/publications/documents/00.reports/00tr013.html>.
4. Lipson, H. F., N. R. Mead, and A. P. Moore. Can We Ever Build Survivable Systems From COTS Components? (CMU/SEI-2001-TN-030). Pittsburg, Penn.: Software Engineering Institute, Carnegie Mellon University, 2001. Available at <www.sei.cmu.edu/publications/documents/01.reports/01tn030.html>.
5. Hevner, A. R., R. C. Linger, G. Walton, and A. Sobel. "The Flow-Service-Quality Framework: Unified Engineering for Large-Scale Adaptive Systems." Proceedings of the Hawaii International Conference on System Sciences-35. Los Alamitos, Calif.: IEEE Computer Society Press, 2002.
6. Vatis, M. A. "Cyber Attacks During the War on Terrorism: A Predictive Analysis." Institute for Security Technology Studies at Dartmouth College, 22 Sept. 2001.
7. Moore, A. P., R. J. Ellison, and R. C. Linger. Attack Modeling for Information Security and Survivability (CMU/SEI-2001-TN-001, ADA388771). Pittsburg, Penn.: Software Engineering Institute, Carnegie Mellon University, 2001. Available at <www.sei.cmu.edu/publications/documents/01.reports/01tn001.html>.


## About the Authors

**Robert Ellison, Ph.D.**, is a senior member of the technical staff in the Software Engineering Institute's Networked Systems Survivability Program. Dr. Ellison's research interests include system survivability and architectural patterns and styles for security architectures. He has a doctorate in mathematics from Purdue University and is a member of the Association for Computing Machinery and International Electrical and Electronics Engineers Computer Society.

**Richard Linger** is a senior member of the technical staff at the Software Engineering Institute's CERT Coordination Center at Carnegie Mellon University (CMU). He teaches at the CMU H. J. Heinz School of Public Policy and Management. Linger has published three software engineering textbooks and more than 50 articles. He has a bachelor's degree in electrical engineering from Duke University. He is member of the International Electrical and Electronics Engineers Computer Society and the National Software Council.

**Howard Lipson, Ph.D.,** has been a computer security researcher at the Software Engineering Institute's (SEI) CERT Coordination Center for 10 years. Dr. Lipson played a major role in extending security research at the SEI into the new realm of survivability, developing many foundational concepts and definitions. Dr. Lipson has been a chair of three International Electrical and Electronics Engineers Information Survivability Workshops. He has a doctorate degree in computer science from Columbia University.

**Nancy Mead, Ph.D.**, is the team leader for the Survivable Systems Engineering team as well as a senior member of the technical staff in the Networked Systems Survivability Program of the Software Engineering Institute (SEI), and a faculty member in the Masters of Software Engineering program at Carnegie Mellon University. She is currently involved in the study of survivable systems requirements and architectures, and the development of professional infrastructure for software engineers. Prior to joining the SEI, Dr. Mead was a senior technical staff member at IBM Federal Systems. She also worked in IBM's software engineering technology area and managed IBM Federal Systems' software engineering education department. She has developed and taught numerous courses on software engineering topics, both at universities and in professional education courses.

**Andrew Moore** is a member of the technical staff at the CERT Coordination Center of Carnegie Mellon University's Software Engineering Institute (SEI). As a participant in the Network Systems Survivability Program, Moore explores ways to improve the survivability of unbounded systems. Before joining the SEI, he worked for the Naval Research Laboratory investigating high-assurance system development methods for the Navy. His research interests include survivable systems engineering, formal assurance techniques, adversary modeling, and security risk analysis. Moore has a bachelor's of arts degree in mathematics from the College of Wooster and a master's of arts degree in computer science from Duke University.


**Software Engineering Institute**
**Carnegie Mellon University**
**4500 Fifth Avenue**
**Pittsburg, PA 15213-3890**